REMARKS

The Examiner is thanked for the telephone interview conducted on July 12, 2007.

By this amendment, Claims 1, 3, 5, 8-11, 14, 17-33, 36, 38-43, and 49-52 are amended, Claims 53-56 are added, and no claims are canceled. Hence, Claims 1-56 are pending in the application.

The amendments to the claims as indicated herein do not add any new matter to this application. Furthermore, amendments made to the claims as indicated herein have been made to exclusively improve readability and clarity of the claims and not for the purpose of overcoming alleged prior art.

Each issue raised in the Final Office Action mailed April 25, 2007 is addressed hereinafter.

I. TELEPHONE INTERVIEW

The Examiner is thanked for the telephone interview conducted on July 12, 2007. In the interview, representatives for the Applicants described some of the fundamental differences between Claim 1 and *Mathur*. Some of the fundamental features of Claim 1 that *Mathur* lacks include software version information, receiving a request for boot images and software packages from a node that is performing an initial boot, and the master node determining, based on software version information of the node, which boot image and software packages from a plurality of boot images and software packages to send to the node. It was the Examiner's position that, although *Mathur* does not teach or suggest software version information, software version information is necessarily used in *Mathur*. No agreement was reached.

II. SUMMARY OF THE OBJECTIONS/ REJECTIONS

Claim 38 is free of any informalities. Removal of the objection with respect to Claim 38 is respectfully requested.

Claims 21-31 are objected to for reciting a computer-readable medium, which may comprise a carrier wave. Claims 21-31 are amended herein to reflect a computer-readable **storage** medium, as suggested in the Final Office Action. Removal of the objection with respect to Claims 21-31 is therefore respectfully requested.

Claims 1, 10, 21, 32, and 43 stand rejected under 35 U.S.C. § 112(2) as allegedly being indefinite for reciting the term "preferred." Amended Claims 1, 10, 21, 32, and 43 no longer include a reference to "preferred." Removal of the 35 U.S.C. § 112(2) rejection with respect to Claims 1, 10, 21, 32, and 43 is therefore respectfully requested.

Claims 1-52 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over U.S. Patent No. 5,008,814 issued to Mathur ("*Mathur*") in view of U.S. Patent No. 6,535,924 issued to Kwok et al. ("*Kwok*"). This rejection is respectfully traversed.

III. ISSUES RELATING TO THE CITED ART

A. CLAIM 1

Claims 1-52 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Mathur* in view of *Kwok*.

According to MPEP § 2143, in order to establish a *prima facie* case of obviousness, "the prior art reference (or references when combined) must teach or suggest **all** the claim limitations" (emphasis added). As will be shown hereinafter, *Mathur* and *Kwok* fail to teach or suggest **all** the limitations of Claim 1.

Present Claim 1 recites:

A method of software loading and initialization in a distributed network of nodes, the method comprising:
persistently storing, in a first storage of a master node, a plurality of software packages and a plurality of boot images, wherein the plurality of software packages and the plurality of boot images will be used by the nodes in the distributed network;
persistently storing, in a second storage of the master node, software version information and node type information for each node in the distributed network;
receiving, at the master node, a request for a boot image and software packages from a node, in the distributed network, that is performing an initial boot;
based on the request, the master node determining software version information of the node to retrieve from the second storage;
the master node retrieving the software version information of the node from the second storage;
the master node determining, based on the software version information of the node, a boot image of the plurality of boot images and one or more software packages of the plurality of software packages to extract from the first storage;
the master node extracting the boot image and the one or more software packages from the first storage;
delivering, to the node, the boot image and the one or more software packages;
wherein the node stores the boot image and the one or more software packages in its local persistent storage;
wherein software version information is extracted from the one or more software packages and stored in the local persistent storage; and
wherein the node reboots and executes the boot image stored in the local persistent storage. (emphasis added)

Mathur and *Kwok*, either individually or in combination, fail to teach or suggest at least the above-bolded features of Claim 1.

As a preliminary matter, it is respectfully noted that the Final Office Action reads claimed features into *Mathur* where *Mathur* has no express disclosure of the features. The Final Office Action's interpretation of *Mathur* is over-broad and reads features into *Mathur* that a skilled artisan would not find in *Mathur*. The Office's practice of using the 'broadest reasonable interpretation' of a claim and a reference does not allow the Office to find in a reference features that are absent.

1. Mathur *fails to teach or suggest persistently storing software version information for each node in a network*

On page 2, the Final Office Action equates the source node (N_s) or operator node (N_o) of *Mathur* with the "master node" of Claim 1. The Final Office Action then asserts that (1) col. 3, line 65 to col. 4, line 49; (2) col. 5, lines 3-30; (3) col. 9, line 36 to col. 10, line 8; and (4) col. 3, lines 29-53 of *Mathur* discloses "storing, in a second storage of the master node, software version information and node type information for each node in the distributed network." This is incorrect.

Nothing in *Mathur* discloses that a source node (N_s) or operator node (N_o) includes a storage that persistently stores **software version information and node type information for each node in the network**. Col. 3, line 65 to col. 4, line 49 describes what contents a non-volatile storage device (NVSD) 103 of a network node may comprise. For example, a NVSD 103 of a node may be classified as "new," "dirty," "trial," or "old," depending on the software contained therein. However, no NVSD 103 persistently stores software version information or node type **for each node in the network**. The claimed features are not inherent in *Mathur* and the Final Office Action provides no evidence that the claimed features are necessarily used in the approach of *Mathur*. A reference that 'almost' teaches claimed features is legally insufficient to support a rejection.

Col. 5, lines 3-30 states that a distribution command is entered at an N_o and that such a command contains a source field and a destination field. The source field identifies a N_s and the location of a NVSD 103 that contains new software. The destination field identifies nodes in the network that will receive the new software. Such nodes are referred to as destination nodes N_d. The destination field also contains location of NVSDs 103 within an N_d that will receive the new software. However, *Mathur* fails to disclose that the N_s or N_o persistently stores software version information and node type **for each node in the network**. Additionally, the distribution

command cannot be equated to the recited "second storage" because the distribution command does not store anything – it is simply a command. Furthermore, even if the distribution command could be considered a persistent storage that stores information, the distribution command does not include **software version information** *and* **node type information for each node in the network**. The distribution command only *identifies* nodes that will receive new system software. The distribution command does not even identify software version information, much less node type information, of a node.

Col. 9, line 36 to col. 10, line 8 describes the general interactions between a master task of the N_s and slave tasks of a N_d. For example, the master task of the N_s requests information from slave tasks. In response, a slave task of a N_d send the requested information (such as which NVSD 103 of the N_d should receive the new software) to the master task. The master task selects destinations for a particular distribution and informs the slave tasks (of selected N_ds) of the number of packets that will be sent and "the identification (version and release number) of the system software to be distributed." Again, nothing in this cited portion of *Mathur* teaches or suggests that the N_s **persistently stores software version information and node type information for each node in the network**.

Col. 3, lines 29-53 of *Mathur* refers to topological information "which is used for communication and routing messages between nodes" (emphasis added). This portion of *Mathur* further states that "network topological information is periodically maintained and updated to reflect changes in the configuration of the network." References to topological information in *Mathur* also fail to teach or suggest persistently storing, in N_s, software version information and node type information for each node in a network.

The Final Office Action, on page 4, further contends that "topology information plus version and node identification in a list being stored in a distinct location for cutback after a

failure in a *softload* reads on a node type and a preferred version information in a trial version to be rolled back at a second storage." However, the only "list" referred to in *Mathur* is a list of destinations in the destination field of a distribution command. There is no list in *Mathur* that is stored for cutback after a failure in a softload process.

2. Mathur *fails to teach or suggest "receiving, at the master node, a request for a boot image and software packages from a node"*

The Final Office Action asserts that the message receiver 303 in FIG. 3 of *Mathur* discloses "receiving, at said master node, a request for a boot image and software packages from a node." This is incorrect.

Col. 10, lines 28-33 of *Mathur* states: "The message receiver 303 receives operator commands 304 and messages 305 from slave tasks in the network. In response to an operator command or a message, the message receiver 303 activates one of a plurality of processes in the message handler block 307." This portion of *Mathur*, as well as the surrounding text, fails to teach or suggest that a request **for a boot image and software packages** is received from a node that is performing an initial boot. Indeed, as indicated above, according to present Claim 1, a node in a network requests a boot image and software packages. In contrast, *Mathur* teaches that "the communication is **initiated by an operator command…to the master task of …the source node N_s** of a softload process" (col. 11, lines 16-21). Col. 4, line 64 to col. 5, line 3 of *Mathur* further states:

> After the new version of system software is installed in the source node N_s, the distribution of the new system software to the other nodes is initiated. Typically, basic operation of the new system software is tested before it is distributed. Advantageously, **the distribution is initiated by a "distribution command" which is entered by the operator at an input/output device in one of the nodes**. (emphasis added)

Therefore, not only does *Mathur* fail to disclose "receiving a request for a boot image and software packages from a node…that is performing an initial boot," *Mathur* **teaches away** from this feature of present Claim 1.

3. Mathur *fails to teach or suggest "based on the request, the master node determining software version information of the node to retrieve from the second storage"*

In the Response to Arguments section (D) of the Final Office Action (on page 16), the Final Office Action asserted that "[t]he *retrieving* limitation is treated as inherent in any process included in the paradigm about delivery of components being stored from the source provider, to the target recipient; hence also disclosed." However, limitations may not be read out of context, i.e., without examining the claim as a whole. It is respectfully submitted that this feature of Claim 1 (i.e., "based on the request, the master node determining software version information of the node to retrieve from the second storage" and "the master node retrieving the software version information of the node from the second storage) cannot be shown by *Mathur* because the source node (N_S) of *Mathur* (1) does not persistently store software version information for each node in a network, (2) does not receive such requests from other nodes, and (3) even if N_S did perform (1) and (2), N_S does not make the recited determination.

The Final Office Action cites (1) col. 5, lines 3-30; (2) col. 9, line 36 to col. 10, line 8; and (3) col. 7, lines 24-44 of *Mathur* for disclosing "based on the request, the master node retrieving software version information of the node from the second storage." This is incorrect.

Col. 5, lines 3-30 and col. 9, line 36 to col. 10, line 8 of *Mathur* have been discussed in detail above with respect to persistently storing software version information in the recited second storage. Both cited portions of *Mathur* also lack any teaching or suggestion of the recited request for a boot image and software packages from a node that is performing an initial boot.

Col. 7, lines 24-44 of *Mathur* describes the checks a destination node (N_d) performs in response to receiving a cutover command after a successful distribution. Specifically, an N_d checks (1) whether the cutover process should proceed and then (2) whether the NVSD 103 of the N_d has a status of "new"; in which case a consistency check is performed on the NVSD 103. If the consistency check is successful, then the status of the NVSD 103 is changed to "trial", after which the N_d performs an initial program load (IPL). This cited portion of *Mathur* also fails to teach or suggest anything relating to retrieving **software version information of the node**, much less retrieving such information based on the recited request.

4. Mathur *fails to teach or suggest "the master node determining, based on the software version information of the node, a boot image of the plurality of boot images and one or more software packages of the plurality of software packages to extract from the first storage"*

In order to teach or suggest this feature of Claim 1, the source node (N_s) of *Mathur* must determine, based on software version information of a particular node, which boot image of a plurality of boot images and which software packages of a plurality of software packages to extract for that node. Even if *Mathur* did teach or suggest that N_s stores software version information for each node in a network (which N_s does not), *Mathur* fails to teach or suggest that N_s makes such a determination. In fact, there is no need to make the determination because a human operator informs N_s, in a distribution command, which nodes to send new software.

Thus, one reason for this lack of teaching is that the invention of *Mathur* depends on (1) a human operator specifying destination nodes (N_d) for a distribution and (2) each destination node containing logic for handling distributions based on a status for each NVSD 103 with which the destination node is associated. In contrast, present Claim 1 recites that a master node can determine (1) software version information (i.e., stored in second storage) of a node based on a request from that node and (2) a particular boot image and set of software packages (i.e., stored

in first storage) based on the determined software version information. There are no similar determinations in *Mathur*.

Based on the foregoing, *Mathur* fails to teach or suggest numerous features of present Claim 1. The Final Office Action does not allege that *Kwok* discloses those features. Therefore, it is respectfully submitted that *Mathur* and *Kwok* fail to teach or suggest (individually or in combination) all the features of present Claim 1. Removal of the 35 U.S.C. § 103(a) rejection with respect to present Claim 1 is respectfully requested.

B. CLAIMS 10, 21, 32, AND 43

Each of independent Claims 10, 21, 32, and 43 is either a method, a computer-readable storage medium, an apparatus, or a system claim. Each of Claims 10, 21, 32, and 43 recite features discussed above that distinguish present Claim 1 over *Mathur* and *Kwok*. Therefore, each of Claims 10, 21, 32, and 43 is allowable for the reasons given above with respect to present Claim 1.

C. CLAIMS 53-56

Each of Claims 53-56 is dependent upon one of the independent claims discussed above, and additionally recites that (1) the request from the node includes node type information of the node and (2) the master node uses the node type information of the node to determine the software version information of the node. Support for Claims 53-56 may be found in paragraph [0081] of the specification.

Mathur and *Kwok* fail to teach or suggest that a request from a destination node (N_d) includes node type information of that node. Therefore, *Mathur* and *Kwok* must also fail to teach or suggest that the source node (N_s) uses the node type information to determine (and ultimately retrieve) software version information of the node from storage.

Based on the foregoing, *Mathur* and *Kwok* fail to teach or suggest (individually or in combination) all the features of Claims 53-56. Removal of the 35 U.S.C. § 103(a) rejection with respect to each of Claims 53-56 is therefore respectfully requested.

D. CLAIMS 3, 20, 31, 42, AND 52

Claims 3, 20, 31, 42, and 52 each recite that functional features are included in the request, from the node, for a boot image and software packages. *Mathur* and *Kwok* fail to teach or suggest that a destination node sends a request (for a boot image and software packages) that includes functional features.

Claims 3, 20, 31, 42, and 52 further recite that the software version information of the node is created based on these functional features. *Mathur* and *Kwok* also fail to teach or suggest that software version information of a node is created based on functional features requested by the same node.

Based on the foregoing, *Mathur* and *Kwok* fail to teach or suggest (individually or in combination) all the features of Claims 3, 20, 31, 42, and 52. Removal of the 35 U.S.C. § 103(a) rejection with respect to each of Claims 3, 20, 31, 42, and 52 is therefore respectfully requested.

E. REMAINING DEPENDENT CLAIMS

The dependent claims not discussed thus far are dependent claims, each of which depends (directly or indirectly) on one of the independent claims discussed above. Each of the dependent claims is therefore allowable for the reasons given above for the claim on which it depends. In addition, each of the dependent claims introduces one or more additional limitations that independently render it patentable. However, due to the fundamental differences already identified, to expedite the positive resolution of this case, a separate discussion of those

limitations is not included at this time. The Applicant reserves the right to further point out the differences between the cited art and the novel features recited in the dependent claims.

III. CONCLUSIONS & MISCELLANEOUS

For the reasons set forth above, all of the pending claims are now in condition for allowance. The Examiner is respectfully requested to contact the undersigned by telephone relating to any issue that would advance examination of the present application.

A petition for extension of time, to the extent necessary to make this reply timely filed, is hereby made. If applicable, a law firm check for the petition for extension of time fee is enclosed herewith. If any applicable fee is missing or insufficient, throughout the pendency of this application, the Commissioner is hereby authorized to any applicable fees and to credit any overpayments to our Deposit Account No. 50-1302.

Respectfully submitted,
HICKMAN PALERMO TRUONG & BECKER LLP

Dated: July 13, 2007

/DanielDLedesma/
Daniel D. Ledesma
Reg. No. 57,181

2055 Gateway Place Suite 550
San Jose, California 95110-1093
Telephone No.: (408) 414-1229
Facsimile No.: (408) 414-1076